ASIAN JOURNAL OF CIVIL ENGINEERING (BUILDING AND HOUSING) VOL. 13, NO. 4 (2012)
PAGES 465-487

OPTIMAL DESIGN OF DOUBLE LAYER BARREL VAULTS USING AN IMPROVED HYBRID BIG BANG-BIG CRUNCH METHOD

A. Kaveh[*] and B. Eftekhar
Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran-16, Iran

Received: 30 September 2011; **Accepted**: 27 December 2011

ABSTRACT

In this paper, optimal design of barrel vaults is performed using the Improved Big Bang-Big Crunch (IBB-BC) method. This method can be used for problems with continuous and discrete variables. Here, double layer braced barrel vaults are optimized considering weight of the structure as the objective function, where the necessary constraints are satisfied. The cross sectional areas are considered as continuous variables.

The algorithm is based on the Big Bang-Big Crunch (BB-BC) and Harmony Search (HS). The BB-BC applies for global optimization and the HS deals with variable constraints. After a number of sequential Big Bangs and Big Crunches, where the distribution of randomness within the search space during the Big Bang becomes smaller and smaller about the average point computed during the Big Crunch, the algorithm converges to a solution. In the proposed method, similar to HPSO, HS is used for controlling the variable constraints.

Keyword: Optimal design; double layer barrel vault; big bang-big crunch; harmony search algorithm

1. INTRODUCTION

The popularity of barrel vaults is partially due to the economy of these structures, since all arches can be constructed as identical hyper-members. At the same time, their cylindrical shape provides a great deal of volume under the roof, a distinct advantage for railway stations, or for large span warehouses, providing a welcome increase in their storage space. Barrel vaults are lightweight and cost effective structures that are used to cover large areas such as exhibition halls, stadium and concert halls. These structures provide a completely unobstructed inner space and they are economical in terms of materials compared to many other conventional forms of structures as explained by Makowski [1]. A barrel vault consists of one or more layers of elements that are arched in one direction. Barrel vaults are given

[*] E-mail address of the corresponding author: alikaveh@iust.ac.ir (A. Kaveh)

different names depending on the way their surface is formed. The earlier types of braced barrel vaults were constructed as single-layer structures. Nowadays, with increase of the spans, double-layer systems are often preferred. If the area to be covered is long compared to its width, and internal stanchions are not permitted, then continuous braced barrel vaults are used. The long barrel vaults are subdivided into several shorter ones, supported internally by stiffening space trusses, integrally interwoven into the framework of the structure [1].

There are many possible ways of bracing which have been used in the construction of single-layer braced barrel vaults. The fully triangulated systems can theoretically be analyzed as pin-jointed structures. The barrel vaults, having the diagonal or hexagonal types of bracing should have rigid joints to be stable, and the influence of bending moments in their stress distribution is much more than in the other types. In order to restrict the length of compression members, especially when the span is wide, a large number of lattices are needed. This will produce nearly collinear lattices, with the consequent danger of instability. However, this can be avoided by using double-layer braced barrel vaults. It has also been shown that the stiffness of very wide barrel vaults can even be increased by interweaving two sets of lattices. In the latter case every other rib is raised above its neighbors [1].

Size optimization of truss structures involves the determination of suitable values for member cross-sectional areas A_i, that minimize the structural weight W. Such a minimum design should also satisfy the inequality constraints that limit design variable sizes and structural responses. The optimal design of a truss can be formulated as [2]:

$$\text{Minimize } W(\{x\}) = \sum_{i=1}^{n} \gamma_i . A_i . L_i \tag{1}$$

Subject to:

$$\begin{gathered}\delta_{\min} \le \delta_i \le \delta_{\max},\ i = 1,2,\ldots,m \\ \sigma_{\min} \le \sigma_i \le \sigma_{\max},\ \ i = 1,2,\ldots,n \\ \sigma_i^b \le \sigma_i \le 0,\ \ i = 1,2,\ldots,ns \\ A_{\min} \le A_i \le A_{\max},\ i = 1,2,\ldots,ng\end{gathered}$$

Where $W(\{x\})$ is the weight of the structure; *n* is the number of members forming the structure; *m* is the number of nodes; *ns* is the number of compression elements; *ng* is the number of groups (number of design variables); γ_i is the material density of member *i*; L_i is the length of member *i*; A_i is the cross-sectional area of member *i* chosen between $\mathrm{A}_{\min}$ and $\mathrm{A}_{\max}$; *min* is the lower bound and *max* is the upper bound; σ_i and δ_i are the stress and nodal deflection, respectively; σ_i^b is the allowable buckling stress in member *i* when it is in compression [2].

In the last decades, different natural evolutionary algorithms have been developed for structural optimization including Genetic Algorithms, Ant Colony Optimization, Particle Swarm Optimizer, Harmony Search, and Charged System Search.

A new optimization method relying on one of the theories of the evolution of the universe, namely the Big Bang and Big Crunch theory, is introduced by Erol and Eksin [3]. The BB–BC algorithm for truss design follows the general procedure developed by Erol and Eksin, but according to the Camp's paper [4]; positions of the candidate solutions at the beginning of each Big Bang are normally distributed around a new point, located between the center of mass and the best global solution.

According to this theory, in the Big Bang phase energy dissipation produces disorder, and randomness is the main characteristic of this phase; while in the Big Crunch phase, the randomly distributed particles are drawn into an order. The Big Bang–Big Crunch (BB–BC) optimization method similarly generates random points in the Big Bang phase and shrinks them into a single representative point via a center of mass in the Big Crunch phase. After a sequence of Big Bangs and Big Crunches, where the distribution of randomness within the search space during the Big Bang becomes smaller and smaller about the average point computed during the Big Crunch, the algorithm converges to a solution [2].

This algorithm not only considers the center of mass as the average point in the beginning of each Big Bang, but also utilizes the best position of each particle and the best visited position of all particles, similar to Particle Swarm Optimization-based approaches [5]. As a result because of increasing the exploration of the algorithm, the performance of the BB–BC approach is improved. Another reformation is to use Sub-Optimization Mechanism (SOM), introduced by Kaveh et al. [6] for ant colony approaches. SOM is based on the principles of finite element method working as a search-space updating technique. Some changes are made to prepare SOM for the HBB–BC algorithm. Another change in the BB-BC algorithm is its combination with the HS algorithm for further improvement. The HS works as a handling approach to deal with variable boundaries. In the IHBB-BC, the HS method employs the HM operator for guiding the exploration. Numerical simulation based on the IHBB–BC method including medium- and large-scale trusses and comparisons with the results obtained by other heuristic approaches, demonstrate the effectiveness of the IHBB-BC.

2. A BRIEF INTRODUCTION TO BB-BC AND HS

In order to make the paper self-explanatory, before proposing the present work for optimal design, the characteristics of BB-BC and HS, are briefly explained in the following two sections:

2.1 Big bang-big crunch optimization

The BB–BC method first developed by Erol and Eksin [3] consists of two phases: a Big Bang phase, and a Big Crunch phase. In the Big Bang phase, candidate solutions are randomly distributed over the search space. Similar to other evolutionary algorithms, initial solutions are spread all over the search space in a uniform manner in the first Big Bang. Erol and Eksin [3] associated the random nature of the Big Bang to energy dissipation or the transformation from an ordered state (a convergent solution) to a disorder or chaos state (new set of solution candidates). The Big Bang phase is followed by the Big Crunch phase. The Big Crunch is a convergence operator that has many inputs but only one output, which is named as the ‘‘center of mass”, since the only output has been derived by calculating the center of mass. Here, the term “mass” refers to the inverse of the merit function value [2].

The BB-BC algorithm can briefly stated as follows:

Step 1: Random formation of some candidates for the initial solution for design variables:

$$\mathrm{A}_i = \mathrm{A}_{\min} + Rand(\mathrm{A}_{\max} - \mathrm{A}_{\min}) \tag{2}$$

Step 2: Analysis of the structure considering the design variables of a candidate solution and evaluation of the penalty functions for each solution candidate.

The penalty function method has been the most popular constraint-handling technique due to its simple principle and ease of implementation. In utilizing the penalty functions, if the constraints are between the allowable limits, the penalty will be zero; otherwise, the amount of penalty is obtained by dividing the violation of allowable limit to the limit itself.

$$\begin{gathered}
\begin{cases}
\sigma_i^{\min} < \sigma_i < \sigma_i^{\max} \Rightarrow \phi_\sigma^{(i)} = 0 \\
\sigma_i^{\min} > \sigma_i or \sigma_i^{\max} < \sigma_i \Rightarrow \phi_\sigma^{(i)} = \dfrac{\sigma_i - \sigma_i^{\min/\max}}{\sigma_i^{\min/\max}}, \; i = 1,2,\dots,n
\end{cases} \\
\begin{cases}
\sigma_b < \sigma_i < 0 \Rightarrow \phi_{\sigma b}^{(i)} = 0 \\
\sigma_i < 0 \wedge \sigma_i < \sigma_b \Rightarrow \phi_{\sigma b}^{(i)} = \dfrac{\sigma_i - \sigma_b}{\sigma_b}, \; i = 1,2,\dots,ns
\end{cases} \\
\begin{cases}
\delta_i^{\min} < \delta_i < \delta_i^{\max} \Rightarrow \phi_\delta^{(i)} = 0 \\
\delta_i^{\min} > \delta_i or \delta_i^{\max} < \delta_i \Rightarrow \phi_\delta^{(i)} = \dfrac{\delta_i - \delta_i^{\min/\max}}{\delta_i^{\min/\max}}, \; i = 1,2,\dots,m
\end{cases}
\end{gathered} \tag{3}$$

Step 3: Calculation of the merit function:

In optimizing structures, the main objective is to find the minimum amount of the merit function. This function is defined as [4]:

$$Mer^k = w^k (1 + \phi_\sigma^k + \phi_\delta^k)^\varepsilon \tag{4}$$

Where ϕ_δ^k is the total deflection penalty for a candidate truss *k* and ϕ_σ^k is the total stress penalty for a candidate truss k, ε = positive penalty exponent. w^k is a function of the weight of candidate truss *k*. ε is set to 1.5 but gradually it is increased to 3 as suggested by Refs. [4,6].

Step 4: Calculation of the center of mass:

$$\mathrm{A}_i^{c(k)} = \frac{\sum_{j=1}^{N} \frac{1}{Mer^j} . \mathrm{A}_i^{(kj)}}{\sum_{j=1}^{N} \frac{1}{Mer^j}}, \quad i = 1,2,\dots,ng \tag{5}$$

Where $\mathrm{A}_i^{(kj)}$ is the *i*th component of the *j*th solution generated in the *k*th iteration; *N* is the population size in Big Bang phase and Mer^j is the merit function for the *j*th candidate [2].

Step 5: Calculation of the new candidates' solution value around the center of mass:

$$\mathrm{A}_i^{(k+1j)} = \mathrm{A}_i^{c(k)} + \frac{r_j\alpha_1(\mathrm{A}_{\max} - \mathrm{A}_{\min})}{k+1}, \quad i = 1,2,...,ng \tag{6}$$

Where r_j is a random number from a standard normal distribution which changes for each candidate, and α_1 is a parameter for limiting the size of the search space [2].

Step 6: Return to step 2 and repeat the algorithm until the condition for the stopping criterion fulfilled. Convergence for each phase is determined when the value of A^{gbest} is not improved for a specified number of analyses.

2.1.1 Hybrid big bang-big crunch algorithm

Camp improved the Big Bang-Big Crunch by using the best global solution in Eq. (7) generating a new population. Selection of A^{gbest} is limited to the solutions that are feasible or designs that have no applied penalty to their structural weight. The final values for the design variables are contained in or decoded from A^{gbest} [4].

$$\mathrm{A}_i^{(k+1j)} = \beta.\mathrm{A}_i^{c(k)} + (1-\beta).\mathrm{A}^{gbest} + \frac{r_j\alpha_1(\mathrm{A}_{\max} - \mathrm{A}_{\min})}{k+1}, \quad i = 1,2,...,ng \tag{7}$$

Where A^{gbest} is the best global solution and β is a parameter controlling the influence of the A^{gbest} on the location of the new candidate solutions.

Kaveh and Talatahari [2] used the particle swarm optimization (PSO) characteristics to improve the performance of the BB-BC algorithm by using of local best and global best solution as following:

$$\begin{aligned}\mathrm{A}_i^{(k+1j)} &= \alpha_2\mathrm{A}_i^{c(k)} + (1-\alpha_2)(\alpha_3\mathrm{A}_i^{gbest(k)} + (1-\alpha_3)\mathrm{A}_i^{lbest(kj)}) \\ &\quad + \frac{r_j\alpha_1(\mathrm{A}_{\max}^{k_{som}} - \mathrm{A}_{\min}^{k_{som}})}{k+1} \left\langle \begin{array}{l} i = 1,2,...,ng \\ j = 1,2,..., N \end{array}\right.\end{aligned} \tag{8}$$

Where $\mathrm{A}_i^{lbest(kj)}$ is the best position of the *j*th particle up to the iteration *k* and $\mathrm{A}_i^{gbest(k)}$ is the best position among all candidates up to the iteration *k*; and α_2 and α_3 are adjustable parameters controlling the influence of the global best and local best on the new position of the candidates, respectively. Using $\alpha_1 = 1$ allows an initial search of the full range of values for each design variable and $\alpha_2 = 0.40$ and $\alpha_3 = 0.80$ are suitable values for the HBB–BC algorithm [2].

Another improvement in the BB–BC method corresponds to employing the Sub-Optimization Mechanism (SOM) as an auxiliary tool which works as a search-space updating mechanism. SOM, based on the principles of finite element method, was first introduced by Kaveh et al. [6]. Similar to the finite element method which requires dividing of the problem domain into many subdomains and using these patches instead of the main domain, SOM divides the search space into sub-domains and performs optimization process into these patches, and then based on the resulted solutions the undesirable parts are deleted, and the remaining space is divided into smaller parts for more investigation in the next stage. This process continues until the remaining space becomes less than the required size for a satisfy accuracy.

The HBB-BC is developed in [2] and is summarized in the following:

Step 1: Calculating cross-sectional area bounds for each group.

If $\mathrm{A}_i^{gbest(k_{som}-1)}$ is the global best solution obtained from the previous stage (k_{som-1}) for design variable *i*, then

$$\begin{cases} \mathrm{A}_{\min,i}^{(k_{som})} = \mathrm{A}_i^{gbest(k_{som}-1)} - \beta_1.(\mathrm{A}_{\max,i}^{(k_{som}-1)} - \mathrm{A}_{\min,i}^{(k_{som}-1)}) \geq \mathrm{A}_{\min,i}^{(k_{som}-1)} \\ \mathrm{A}_{\max,i}^{(k_{som})} = \mathrm{A}_i^{gbest(k_{som}-1)} + \beta_1.(\mathrm{A}_{\max,i}^{(k_{som}-1)} - \mathrm{A}_{\min,i}^{(k_{som}-1)}) \leq \mathrm{A}_{\max,i}^{(k_{som}-1)} \end{cases} \Big/ \begin{matrix} i = 1,2,...,ng \\ k_{som} = 2,...,nc \end{matrix} \tag{9}$$

Where β_1 is an adjustable factor which determines the amount of the remaining search space and in this research it is taken as 0.3, Ref. [2]; $\mathrm{A}_{\min,i}^{(k_{som})}$ and $\mathrm{A}_{\max,i}^{(k_{som})}$ are the minimum and the maximum allowable cross-sectional areas at the stage (k_{som}), respectively. In stage 1, the amounts of $\mathrm{A}_{\min,i}^{(1)}$ and $\mathrm{A}_{\max,i}^{(1)}$ are set to:

$$\mathrm{A}_{\min,i}^{(1)} = \mathrm{A}_{\min}, \mathrm{A}_{\max,i}^{(1)} = \mathrm{A}_{\max}, \quad i = 1,2,...,ng \tag{10}$$

Step 2: Determining the amount of the increment in the allowable cross sectional areas.

In each stage, the number of permissible value for each group is considered as β_2 , and thus the amount of the accuracy rate of each variable is equal to:

$$\mathrm{A}_i^{*(k_{som})} = \frac{(\mathrm{A}_{\max,i}^{(k_{som})} - \mathrm{A}_{\min,i}^{(k_{som})})}{\beta_2 - 1}, \quad i = 1,2,...,ng \tag{11}$$

Where $\mathrm{A}_i^{*(k_{som})}$ is the amount of increment in the allowable cross-sectional area. Unlike ACO, β_2 (the number of subdomains) does no affect the optimization time and in the BB–BC optimization, β_2 is set to 100 as in Ref. [2].

Step 3: Creating the series of the allowable cross-sectional areas.

The set of allowable cross-sectional areas for group *i* can be defined as

$$\mathrm{A}_{\min,i}^{(k_{som})}, \mathrm{A}_{\min,i}^{(k_{som})} + \mathrm{A}_i^{*(k_{som})}, ..., \mathrm{A}_{\min,i}^{(k_{som})} + (\beta_2 - 1).\mathrm{A}_i^{*(k_{som})} = \mathrm{A}_{\max,i}^{(k_{som})}, \ i = 1,2,...,ng \tag{12}$$

Step 4: Determining the optimum solution of the stage (k_{som}).
The last step is performing an optimization process using the BB–BC algorithm.

Step 5: The stopping creation for the SOM can be described as:

$$\mathrm{A}_i^{*(nc)} \leq \mathrm{A}^*, \quad i = 1,2,...,ng \tag{13}$$

Where $\mathrm{A}_i^{*(nc)}$ the amount of accuracy rate of the last stage; and A^* = the amount of accuracy rate of the primary problem.

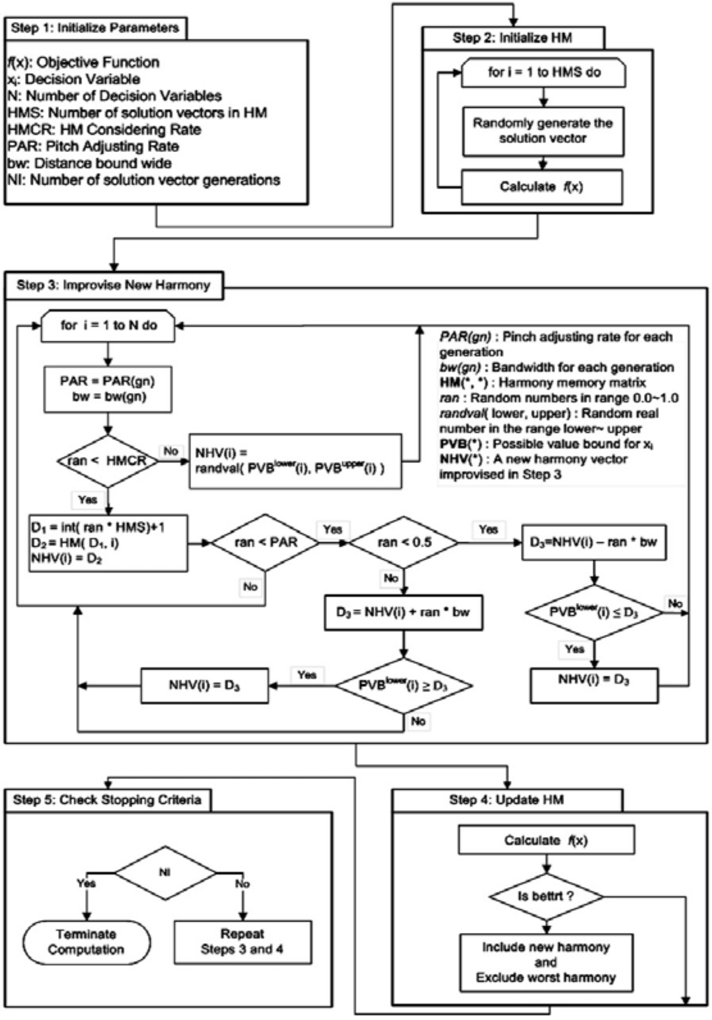
Step 1: Initialize Parameters
f(x): Objective Function
x_i: Decision Variable
N: Number of Decision Variables
HMS: Number of solution vectors in HM
HMCR: HM Considering Rate
PAR: Pitch Adjusting Rate
bw: Distance bound wide
NI: Number of solution vector generations
Step 2: Initialize HM
for i = 1 to HMS do
Randomly generate the solution vector
Calculate f(x)
Step 3: Improvise New Harmony
for i = 1 to N do
PAR = PAR(gn)
bw = bw(gn)
PAR(gn) : Pinch adjusting rate for each generation
bw(gn) : Bandwidth for each generation
HM(*, *) : Harmony memory matrix
ran : Random numbers in range 0.0~1.0
randval(lower, upper) : Random real number in the range lower~ upper
PVB(*) : Possible value bound for x_i
NHV(*) : A new harmony vector improvised in Step 3
ran < HMCR
No
NHV(i) = randval(PVB^lower(i), PVB^upper(i))
Yes
D_1 = int(ran * HMS)+1
D_2 = HM(D_1, i)
NHV(i) = D_2
ran < PAR
Yes
No
ran < 0.5
Yes
No
D_3=NHV(i) – ran * bw
D_3 = NHV(i) + ran * bw
PVB^lower(i) ≤ D_3
No
Yes
NHV(i) = D_3
PVB^lower(i) ≥ D_3
Yes
No
NHV(i) = D_3
Step 5: Check Stopping Criteria
NI
Yes
No
Terminate Computation
Repeat Steps 3 and 4
Step 4: Update HM
Calculate f(x)
Is bettrt ?
Include new harmony and Exclude worst harmony

Figure 1. Optimization procedure of the improved harmony search algorithm [10]

2.2 Harmony search algorithm for continuous variable

Harmony search algorithm was recently developed in an analogy with music improvisation process, where music players improvise the pitches of their instruments to obtain better harmony. The steps in the procedure of harmony search are shown in Figure 1. These steps are as follows [9]:

Step 1. Initialize the problem and algorithm parameters.
Step 2. Initialize the harmony memory.
Step 3. Improvise a new harmony.
Step 4. Update the harmony memory.
Step5. Check the stopping criterion.

2.2.1 Initialize the problem and algorithm parameters

In Step 1, the optimization problem is defined as follows:

Minimize $f(x)$ subject to

$$x_i \in X_i \qquad i = 1,2,...,N. \tag{14}$$

Where $f(x)$ is the objective function; X is the set of each continuous variable x_i; N is the number of variables, X_i is the set of the possible range of values for each variable, that is $x_i^L \le X_i \le x_i^U$, x_i^L and x_i^U are the lower and upper bounds for each continuous variable, respectively. The HS algorithm parameters are also specified at this step. These are the harmony memory size (HMS), or the number of solution vectors in the harmony memory; harmony memory considering rate (HMCR); pitch adjusting rate (PAR); and the number of improvisations (NI), or stopping criterion.

The harmony memory (HM) is a memory location where all the solution vectors (sets of variables) are stored. Here, HMCR and PAR are parameters that are utilized to improve the solution vector. Both are defined in Step 3.

2.2.2 Initialize the harmony memory

In Step 2, the HM matrix is filled with as many randomly generated solution vectors as the HMS

$$[H] = \begin{bmatrix} x_1^1 & x_2^1 & \ldots\ .. & x_{n-1}^1 & x_n^1 \\ x_1^2 & x_2^2 & ..\ .. & x_{n-1}^2 & x_n^2 \\ .. & .. & ..\ .. & .. & .. \\ .. & .. & ..\ .. & .. & .. \\ x_1^{hms-1} & x_2^{hms-1} & ..\ .. & x_{n-1}^{hms-1} & x_n^{hms-1} \\ x_1^{hms} & x_2^{hms} & ..\ .. & x_{n-1}^{hms} & x_n^{hms} \end{bmatrix} \tag{15}$$

2.2.3 Improvise a new harmony

A new harmony vector $x^i = \left(x_1^i, x_2^i, ..., x_n^i\right)$ is generated based on three rules: (1) memory consideration, (2) pitch adjustment and (3) random selection. Generating a new harmony is called 'improvisation' [9].

In the memory consideration, the value of the first variable $\left(x_1^i\right)$ for the new vector is chosen from any of the values in the specified HM range $\left(x_1^1, x_1^2, ..., x_1^{hms}\right)$. Values of the other continuous variables $\left(x_2^i, x_3^i, ..., x_n^i\right)$ are chosen in the same manner. The HMCR, which varies between 0 and 1, is the rate of choosing one value from the historical values stored in the HM, while (1 − HMCR) is the rate of randomly selecting one value from the possible range of values.

$$x_i^{new} = \begin{cases} x_i \quad \in \quad \left\{x_i^1, x_i^2, ..., x_i^{hms}\right\} \quad with\ \ probability\ HMCR \\ x_i \quad \in \quad \left\{x_1, x_2, ..., x_n\right\} \quad with\ \ probability\ (1 - HMCR) \end{cases} \tag{16}$$

For example, a *HMCR* of 0.85 indicates that the HS algorithm will choose the continuous variable value from historically stored values in the HM with a probability of 85% or from the entire possible range with a probability (100–85) %. Every component obtained by the memory consideration is examined to determine whether it should be pitch-adjusted. This operation uses the PAR parameter, which is the rate of pitch adjustment as follows:

$$\text{Is } x_i^{new} \text{ to be pitch-adjusted?} \begin{cases} Yes \quad with\ probability\ of\ \ PAR \\ No \quad with\ probability\ of\ \ (1 - PAR) \end{cases} \tag{17}$$

The value of (1−PAR) sets the rate of doing nothing. If the pitch adjustment decision for x_i^{new} is YES, then x_i^{new} is replaced as follow:

$$x_i^{new} \leftarrow x_i^{new} \pm rand * bw, \tag{18}$$

Where *bw* is an arbitrary distance bandwidth and *rand* is a random number between 0 and 1.

In Step 3, HM consideration, pitch adjustment or random selection is applied to each variable of the new harmony vector in turn.

2.2.4 Update harmony memory

If the new harmony vector is better than the worst harmony in the HM, judged in terms of the objective function value, the new harmony is included in the HM and the existing worst harmony is excluded from the HM. If the new harmony vector is severely infeasible, it is discarded. If it is slightly infeasible, there are two ways to follow. One is to include them in the harmony memory matrix by imposing a penalty on their objective function value. In this way the violated harmony vector which may be infeasible slightly in one or more constraints, is used as a base in the pitch adjustment operation to provide a new harmony vector that may be feasible. The other way is to use larger error values such as 0.08 initially for the acceptability of the new design vectors and reduce this value gradually during the design cycles and use finally an error value of 0.001 towards the end of the iterations. This adaptive error strategy is employed in the design examples of this paper.

2.2.5 Check stopping criterion

If the stopping criterion (maximum number of improvisations) is satisfied, computation is terminated. Otherwise, Steps 3 and 4 are repeated.

3. AN IMPROVE HYBRID BIG BANG BIG CRUNCH FOR CONTINUES VARIABLE

The process of heuristic Big Bang-Big Crunch harmony search optimization algorithm is prepared in Figure 2. This algorithm applies BB-BC for global optimization, while HS works as the operator for preventing the violation of the variable constraints, wherein, HS apply harmony memory-guided mechanism to rectify the positions found by particles in the BB-BC stage.

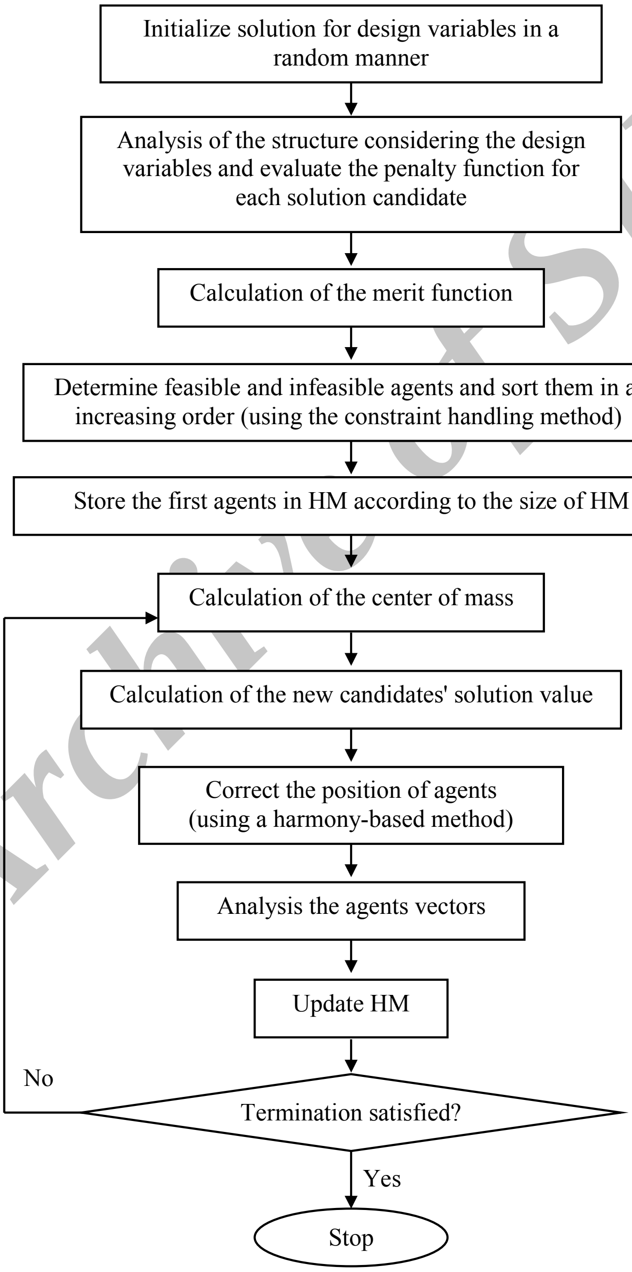
Initialize solution for design variables in a random manner
Analysis of the structure considering the design variables and evaluate the penalty function for each solution candidate
Calculation of the merit function
Determine feasible and infeasible agents and sort them in an increasing order (using the constraint handling method)
Store the first agents in HM according to the size of HM
Calculation of the center of mass
Calculation of the new candidates' solution value
Correct the position of agents (using a harmony-based method)
Analysis the agents vectors
Update HM
No
Termination satisfied?
Yes
Stop

Figure 2. The flowchart of the present method

4. DESIGN EXAMPLES

In this section, first the IHBBBC-based algorithm is tested for the 25-bar truss structure with conditions being defined. Then two double layer barrel vaults are optimized and the results are compared through three methods HS, BB-BC, and IHBB-BC.

The second example is a double barrel vault taken from Ref. [1]. It is optimized under two types of symmetrical and non-symmetrical loadings. The third example is chosen as a double layer braced barrel vault with 984 elements having a large span and a short length. It is optimized by the above mentioned three methods.

In last two examples the size of population is selected as 100, and $A^* = 0.01$. Modulus of elasticity is taken as 30450ksi (210000MPa) and the material density is 0.228 lb/in^3 (7971.810 kg/m^3). The yield stress of steel is taken as 58ksi (400MPa). The minimum cross-sectional area of all members is 0.775 in^2 (2 cm^2) and the maximum cross- sectional area is taken as 20 in^2 (129.03 cm^2). Displacement limitations of ±0.1969 in (5mm) are imposed on all the nodes in x, y and z directions. According to the AISC ASD code [7], the limitations of the stresses are as follows:

$$\begin{cases} \sigma_i^+ = 0.6F_y & for\ \sigma_i \geq 0 \\ \sigma_i^- & for\ \sigma_i < 0 \end{cases} \tag{19}$$

Where σ_i^- is calculated according to the slenderness ratio:

$$\begin{cases} \left[(1 - \frac{\lambda_i^2}{2C_c^2})F_y \right] / (\frac{5}{3} + \frac{3\lambda_i}{8C_c} - \frac{3\lambda_i^3}{8C_c^3}) & for\ \lambda_i < C_c \\ \frac{12\pi^2 E}{23\lambda_i^2} & for\ \lambda_i \geq C_c \end{cases} \tag{20}$$

Here, E is the modulus of elasticity; F_y is the yield stress of steel; C_c is the slenderness ratio (λ_i) dividing the elastic and inelastic buckling regions ($C_c = \sqrt{(2\pi^2 E)/F_y}$); λ_i is the slenderness ratio ($\lambda_i = kL_i / r_i$); k is the effective length factor; L_i = the member length; and r_i is the radius of gyration.

On the other hand, the radius of gyration (r_i) can be expressed in terms of cross-sectional areas, i.e. aA_i^b. Here, a and b are the constants depending on the types of sections adopted for the members such as pipes, angles, and tees. In this example, pipe sections (a = 0.4993 and b = 0.6777) are adopted for bars similar to that of [8].

In addition, the maximum slenderness ratio is limited to 300 for tension members, and it is recommended to be 200 for the compression members according to ASD-AISC [7] design code provisions which can be formulated as follows:

$$\begin{cases} \lambda_m = \dfrac{k_m l_m}{r_m} \leq 300 & \textit{for tension members} \\ \lambda_m = \dfrac{k_m l_m}{r_m} \leq 200 & \textit{for compression members} \end{cases} \tag{21}$$

Where k_m is the effective length factor of the *m*th member ($k_m = 1$ for all truss members), and r_m is its minimum radius of gyration.

All examples in this study are analyzed using direct stiffness method. To show the randomness of these experiments, they are performed for several times and the best results are chosen and shown in the tables.

Example 1: A 25-bar spatial truss

The topology and nodal numbering of a 25-bar spatial truss structure are shown in Figure 3. In this example, designs for a multiple load case are performed and the results are compared to those of other optimization techniques. In these studies, the material density is considered as 0.1 lb/in^3 (2767.990 kg/m^3) and the modulus of elasticity is taken as 10,000 ksi (68,950 MPa). Twenty five members are categorized into eight groups, as follows:

(1) A_1, (2) $A_2 - A_5$, (3) $A_6 - A_9$, (4) $A_{10} - A_{11}$, (5) $A_{12} - A_{13}$, (6) $A_{14} - A_{17}$, (7) $A_{18} - A_{21}$, (8) $A_{22} - A_{25}$.

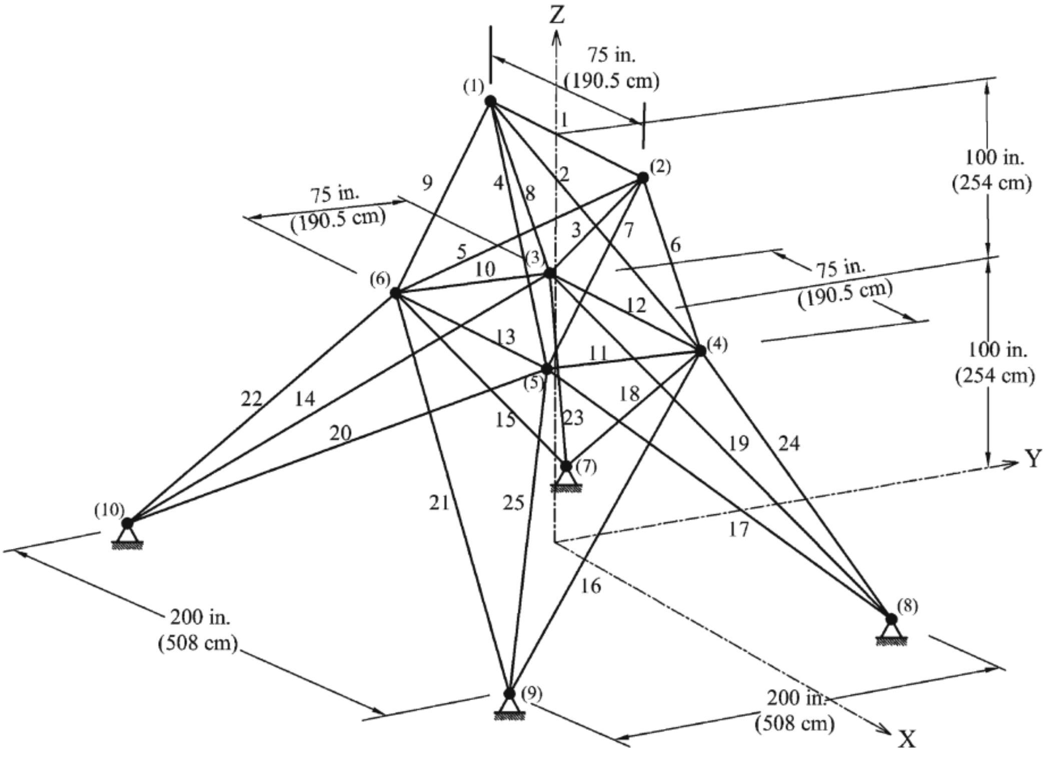

Figure 3. A twenty five-bar spatial truss

This spatial truss is subjected to two loading conditions shown in Table 1. Maximum displacement limitations of ± 0.35 in (8.89 mm) are imposed on every node in every direction and the axial stress constraints vary for each group as shown in Table 2. The range of the cross-sectional areas varies from 0.01 to 3.4 in^2 (0.6452–21.94 cm^2).

Table 1: Loading conditions for the 25-bar spatial truss

Node	Case 1			Case 2		
1	0.0	20.0 (89)	–5.0 (22.25)	1.0 (4.45)	10.0 (44.5)	–5.0 (22.25)
2	0.0	–20.0 (89)	–5.0 (22.25)	0.0	10.0 (44.5)	–5.0 (22.25)
3	0.0	0.0	0.0	0.5 (2.22)	0.0	0.0
6	0.0	0.0	0.0	0.5 (2.22)	0.0	0.0

Table 2: Member stress limitation for the 25-bar spatial truss

Element group		Compressive stress limitations ksi (MPa)	Tensile stress limitations ksi (MPa)
1	A_1	35.092 (241.96)	40.0 (275.80)
2	$A_2 \sim A_5$	11.590 (79.913)	40.0 (275.80)
3	$A_6 \sim A_9$	17.305 (119.31)	40.0 (275.80)
4	$A_{10} \sim A_{11}$	35.092 (241.96)	40.0 (275.80)
5	$A_{12} \sim A_{13}$	35.092 (241.96)	40.0 (275.80)
6	$A_{14} \sim A_{17}$	6.759 (46.603)	40.0 (275.80)
7	$A_{18} \sim A_{21}$	6.759 (46.603)	40.0 (275.80)
8	$A_{22} \sim A_{25}$	11.082 (76.410)	40.0 (275.80)

The IHBB–BC algorithm achieves the best solution after 5500 searches. However, the CSS algorithm finds the best solution after about 7000 analyses [13] which is 27.3% more than the present work. The best weight of the IHBB–BC is 545.07 lb while the best result of the CSS is 545.10 lb. In addition, the IHBB–BC algorithm has better performance than the CSS algorithm with regard to the average weight. Although the IHBB–BC approach has worse performance than the improved methods (IACS [6] and HPSACO [11]), it performs is better than the other simple algorithms (similar PSO [12]) when the best weight and the average weight are compared.

Also, the IHBB–BC approach has smaller required number of iterations for convergence than HPSACO and HS [9]. Table 3 presents a comparison of the performance of the IHBB–BC method and other heuristic algorithms. The results of the barrel vault of Example 1 under symmetric and unsymmetrical loadings are provided in Table 4 and table 5, respectively.

Table 3: Performance comparison for the 25-bar spatial truss

Element group		Optimal cross-sectional areas *(in^2)*							
		Schutte and Groenwold [12] PSO	Lee and Geem [9] HS	Kaveh et al. [6] IACS	Kaveh and Talatahari			Present work	
					[11] HPSACO	[2] HBB-BC	[13] CSS	*(in^2)*	*(cm^2)*
1	A_1	(2003) PSO	0.047	0.010	0.010	0.010	0.010	0.010	0.065
2	$A_2 \sim A_5$	2.121	2.022	2.042	2.054	1.993	2.003	1.979	12.863
3	$A_6 \sim A_9$	2.893	2.950	3.001	3.008	3.056	3.007	3.001	19.506
4	$A_{10} \sim A_{11}$	0.010	0.010	0.010	0.010	0.010	0.010	0.010	0.065
5	$A_{12} \sim A_{13}$	0.010	0.014	0.010	0.010	0.010	0.010	0.010	0.065
6	$A_{14} \sim A_{17}$	0.671	0.688	0.684	0.679	0.665	0.687	0.687	4.465
7	$A_{18} \sim A_{21}$	1.611	1.657	1.625	1.611	1.642	1.655	1.680	10.92
8	$A_{22} \sim A_{25}$	2.717	2.663	2.672	2.678	2.679	2.660	2.654	17.251
Best weight (lb)		545.21	544.38	545.03	544.99	545.16	545.10	545.07	2424.56
Average weight (lb)		546.84	N/A	545.74	545.52	545.66	545.58	545.49	2426.43
No. of analyses		9596	15000	3520	9875	12500	7000	5500	

Table 4: The results of the barrel vault of Example 1 under symmetric loading

Element group	Optimal cross-sectional areas (*in*^{2})			
	HS		BB-BC	Present work
			(*in*^{2})	(*cm*^{2})
1	0.786	0.817	0.775	4.999
2	1.181	1.201	1.048	6.761
3	1.130	1.253	1.399	9.026
4	0.778	0.775	0.775	4.999
5	6.280	5.753	6.523	42.084
6	0.776	0.776	0.775	4.999
7	14.917	14.392	13.288	85.729
8	10.460	10.403	10.352	66.787
9	15.972	15.906	14.825	95.645
10	16.133	14.868	15.349	99.025
11	11.067	9.714	10.219	65.929
12	12.681	14.540	13.747	88.690
13	7.058	6.853	7.033	45.374
14	4.303	4.483	4.730	30.516
15	2.398	2.449	2.497	16.110
16	4.504	4.299	5.030	32.451
17	6.214	6.289	6.692	43.174
18	0.782	0.775	0.775	4.999
19	0.789	0.790	0.775	4.999
20	0.776	0.789	0.775	4.999
21	0.829	0.793	0.775	4.999
22	0.787	0.775	0.775	4.999
23	0.782	0.775	0.775	4.999
24	2.375	3.073	3.011	19.425
25	1.458	1.502	1.811	11.684
26	1.722	1.617	1.732	11.174
27	2.368	2.757	2.824	18.219
28	1.242	1.278	1.217	7.852
29	1.269	1.419	1.279	8.251
30	1.327	1.252	1.255	8.097
31	1.227	1.229	1.231	7.942
Best weight (lb)	62206	62096	61972	275696 N
Average weight (lb)	62539	62532	62196	276692 N
No. of analyses	50000	50000	50000	

Table 5: The results of the barrel vault of example 1 under unsymmetrical loading

Element group	Optimal cross-sectional areas (in^2)			
	HS	BB-BC	Present work	
			(in^2)	(cm^2)
1	0.775	0.775	0.775	4.999
2	0.823	0.868	0.775	4.999
3	2.442	1.891	0.829	5.348
4	1.780	2.077	2.151	13.877
5	3.006	2.356	2.544	16.413
6	0.918	0.775	0.775	4.999
7	7.768	7.657	7.842	50.593
8	8.002	7.564	7.717	49.787
9	7.768	7.378	7.714	49.767
10	4.833	5.704	5.128	33.083
11	3.286	3.286	3.284	21.187
12	5.024	4.774	4.957	31.980
13	1.709	1.612	1.542	9.948
14	1.435	1.426	1.397	9.013
15	1.936	1.984	1.907	12.303
16	5.371	5.332	5.053	32.600
17	5.082	5.239	5.149	33.219
18	0.775	0.775	0.775	4.999
19	0.775	0.775	0.775	4.999
20	1.537	1.147	0.775	4.999
21	0.780	0.775	0.775	4.999
22	0.775	0.775	0.775	4.999
23	0.775	0.775	0.775	4.999
24	0.799	0.868	0.784	5.058
25	0.775	0.775	0.775	4.999
26	1.373	1.240	1.283	8.277
27	1.960	2.077	2.169	13.993
28	0.775	0.775	0.775	4.999
29	0.975	1.054	0.960	6.193
30	1.621	1.798	1.774	11.445
31	1.160	1.240	1.159	7.477
Best weight (lb)	35501	35372	34731	154508 N
Average weight (lb)	37071	36867	35647	158583 N
No. of analyses	50000	50000	50000	

Example 2: A three hundred eighty four-bar double layer barrel vault

In this example, there are two types of loadings on the structure, and the structure is optimized using three methods HS, BB-BC and IHBB-BC. The structure consists of two rectangular nets and for making it stable, angles of the bottom nets are put into the center of one of the above nets, and these are connected through diametrical elements as shown in Figures 4 and 5.

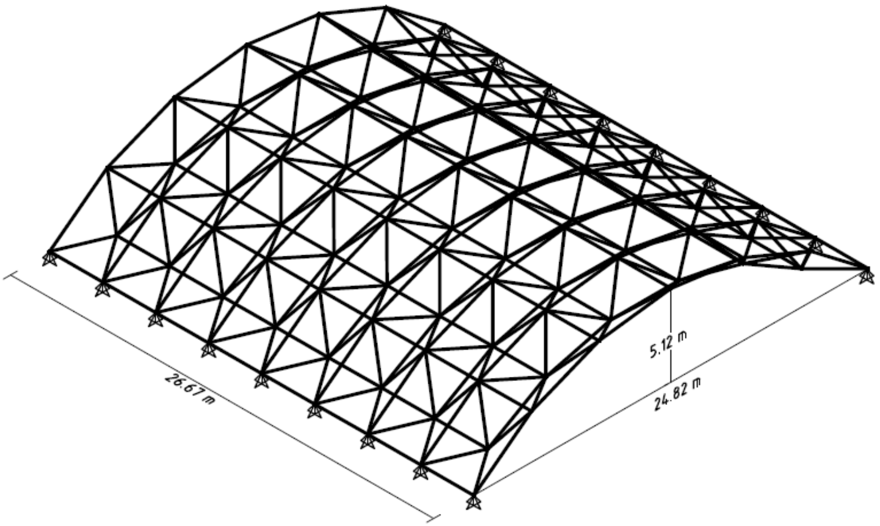

Figure 4. The 3D view of a double layer barrel vault [1]

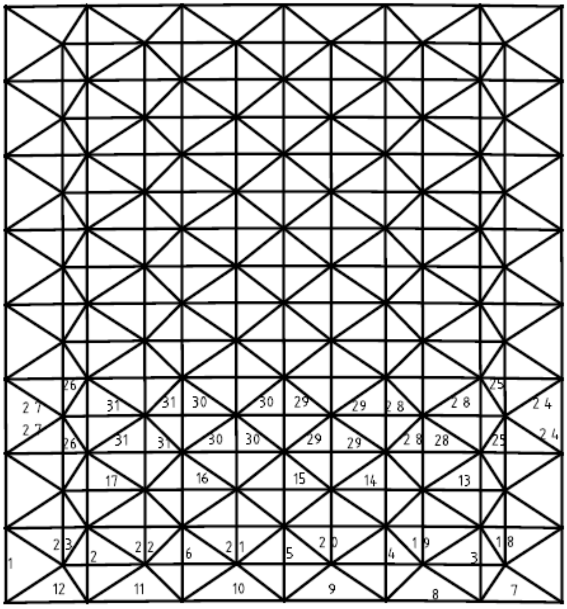

Figure 5. Top view of a square on diagonal double layer barrel vault

For the first loading, which is symmetric, the vertical loads of magnitude −20 kips (−88.968kN) are applied at free joints (non-support joints). For the second loading, which is an unsymmetrical one, concentrated loads of magnitude −10ksi (−44.484kN) are applied at the right hand half of the structure and on non-supported joints. At the left hand half of the structure and on the non-supported joints the load is equal to −6ksi (−29.69kN). According to Ref. [1], the supports are considered at the two external edges of the top layer of the barrel vault and the results are provided in Tables 4 and 5. It can be observed that at the same number of iterations, the result obtained by IHBB-BC is better than the other two algorithms. The convergence rate of the IHBB-BC is higher than BB-BC and HS.

Using IHBB-BC, better results are obtained in comparison with the other two approaches This can be observed from the convergence diagrams of Figures 6 & 7.

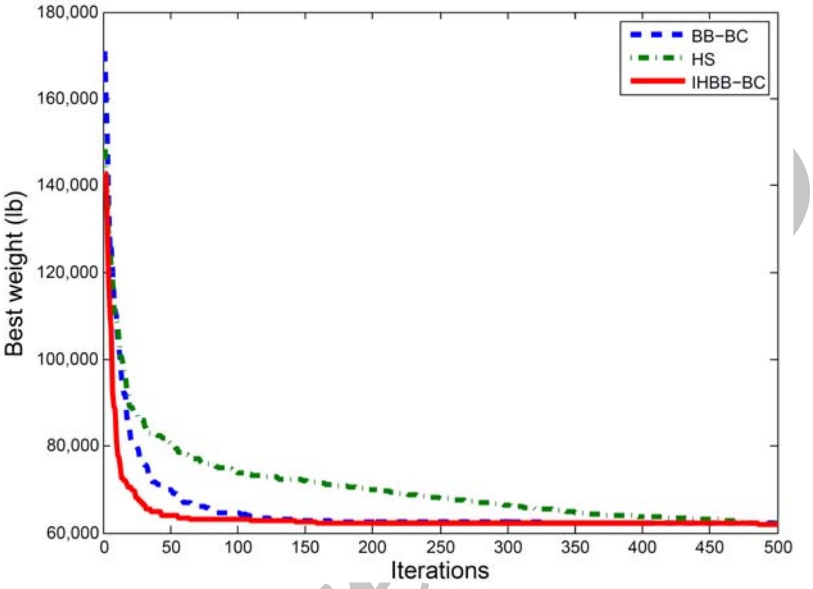

Figure 6. Comparison of the convergence history of the double layer barrel vault for three algorithms under symmetric loading

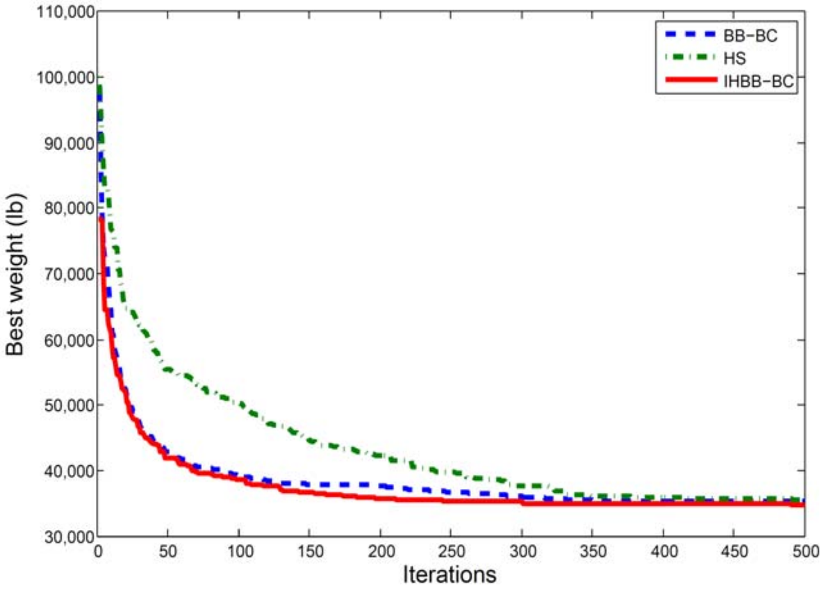

Figure 7. Comparison of the convergence history of double layer barrel vault for three algorithms under unsymmetrical loading

It can also be seen that the members between the supports do not have forces and therefore the smallest cross sections are allocated to these members. Tables 4 and 5 for the members of group 1 have the smallest cross section areas showing a good performance of the structure. In Figure 8 the diagram for the stress ratio ($\frac{\left|\sigma_i^{\max}\right|}{\left|\sigma_i^{allowable}\right|}$) of the maximum stress in the element groups to the maximum allowable stress is depicted. This ratio for some groups is zero since they sustain either no axial force or this force is small. For these members the smallest cross sections are selected. As an example, one can refer to member groups 18 to 23. In investigating the process of optimization we also observe that some member groups have high axial force and high displacement where the constraints corresponding to the displacements or the compressive buckling load have governed the design. For these member groups, wit the axial force governs the design, a suitable cross section with the stress ratio close to 1 are selected.

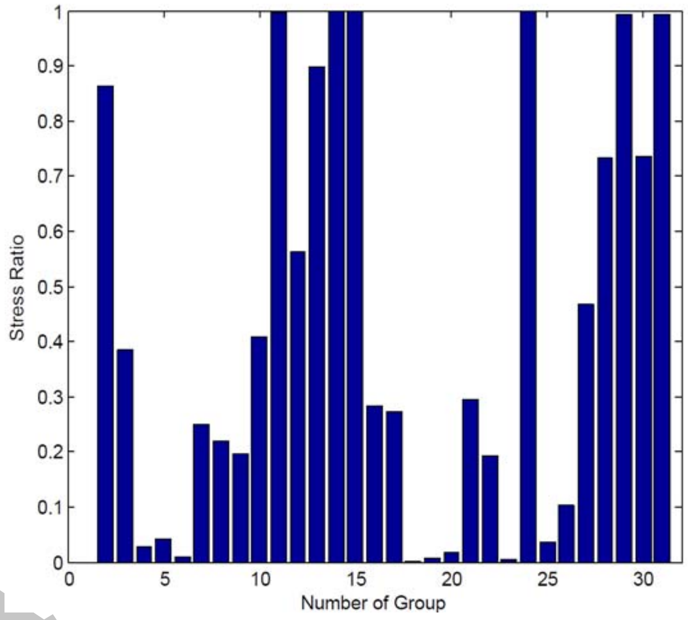

Figure 8. The ratio of the max stress to the allowable stress for different group for the double layer barrel vault under unsymmetrical loading using the IHBB-BC algorithm

Example 3: A 984-bar double layer braced truss type barrel vault

The considered structure is a braced barrel vault as shown in Figure 9 and the loading consists of the following:

1. In the nodes of the central arc, a downward concentrated load of −15 kips (−66.726kN).
2. In the nodes of the arcs adjacent to the central arc, a downward concentrated load of −10 kips (−44.484kN).
3. In the nodes of the arcs adjacent to the external arcs, a downward concentrated load of −5 kips (−22.242kN).

4. In the nodes of the external central arcs, a downward concentrated load of −2 kips (−8.897kN).

This structure consists of 984 elements and considering the type of loading, 55 element groups are selected. The related details are shown in Figures 9 & 10. As it can be observed from Table 6 and the convergence diagram of Figure 11, for the same number of iterations among the three applied methods, IHBB-BC has achieved the best result. The weight achieved by IHBB-BC is 72440lb while for BB-BC and HS, the weights are 74390 and 800021, respectively. During the optimization process the weight of the structure for the IHBB-BC has been smaller than the other two approaches.

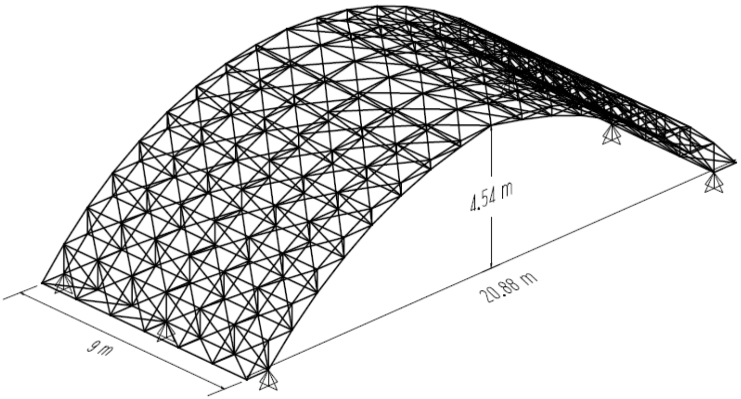

Figure 9. The 3D view of a double layer braced barrel vault

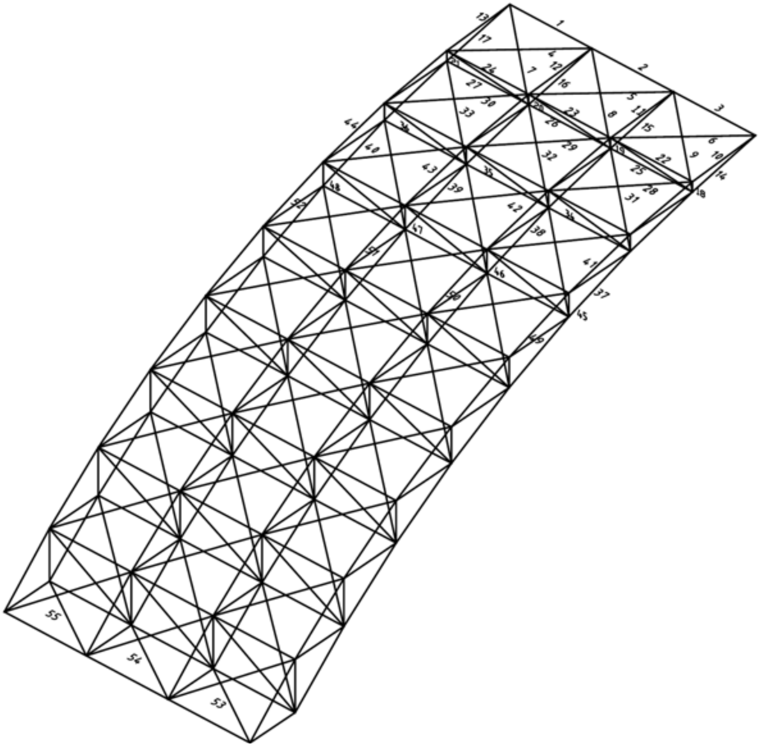

Figure 10. A quarter of the double layer braced barrel vault with the related member grouping

Table 6: The results for the barrel vault of Example 2 under symmetric loading

Element group	**Optimal cross-sectional areas (*in*^{2})**			
	HS	**BB-BC**	**Present work**	
			(*in*^{2})	**(*cm*^{2})**
1	14.365	18.151	1.221	7.877
2	1.499	8.159	3.535	22.806
3	12.315	7.855	2.543	16.406
4	11.871	7.945	1.711	11.038
5	13.539	0.776	0.779	5.026
6	8.649	12.744	0.804	5.187
7	4.181	10.488	1.499	9.671
8	2.708	1.012	0.890	5.742
9	8.155	7.790	1.286	8.297
10	14.288	7.920	3.640	23.484
11	9.169	6.853	9.540	61.548
12	19.146	16.848	18.899	121.928
13	18.723	18.995	17.913	115.567
14	14.174	1.436	2.890	18.645
15	1.462	7.181	2.669	17.219
16	6.078	16.069	0.781	5.038
17	14.909	0.951	1.523	9.826
18	6.158	11.486	2.836	18.297
19	3.420	13.164	8.369	53.993
20	16.034	15.077	11.286	72.812
21	11.776	13.459	5.278	34.051
22	1.010	1.039	0.984	6.348
23	1.228	1.221	1.052	6.787
24	0.775	0.775	2.181	14.071
25	5.249	4.218	5.468	35.277
26	2.779	2.517	3.251	20.974
27	2.243	2.427	2.422	15.625
28	0.775	0.775	0.778	5.019
29	0.835	0.826	0.780	5.032
30	1.387	1.581	1.385	8.935
31	11.626	11.134	13.158	84.890
32	5.945	5.508	6.520	42.064
33	1.266	0.957	0.797	5.142
34	1.260	1.288	1.224	7.896
35	1.108	0.998	1.296	8.361
36	1.713	1.746	1.804	11.638
37	1.240	1.170	2.827	18.238
38	3.120	3.190	2.804	18.090
39	12.690	13.893	9.431	60.845
40	18.198	18.869	18.253	117.761
41	16.336	12.235	17.121	110.458
42	9.843	13.449	8.569	55.283
43	15.597	15.270	17.649	113.864
44	20.00	19.966	19.291	124.458
45	1.188	0.855	0.781	5.038
46	1.090	1.774	1.385	8.935
47	8.687	1.636	4.076	26.296
48	1.314	1.337	5.284	34.090
49	0.887	1.087	1.570	10.129
50	2.218	2.220	2.509	16.187
51	8.156	6.195	6.729	43.413
52	8.943	8.671	7.914	51.058
53	18.889	10.397	1.539	9.929
54	0.873	0.975	0.775	4.999
55	9.188	1.023	0.786	5.071
Best weight (lb)	80021	74390	72440	322265 N
Average weight (lb)	80616	75285	73702	327879 N
No. of analyses	50000	50000	50000	

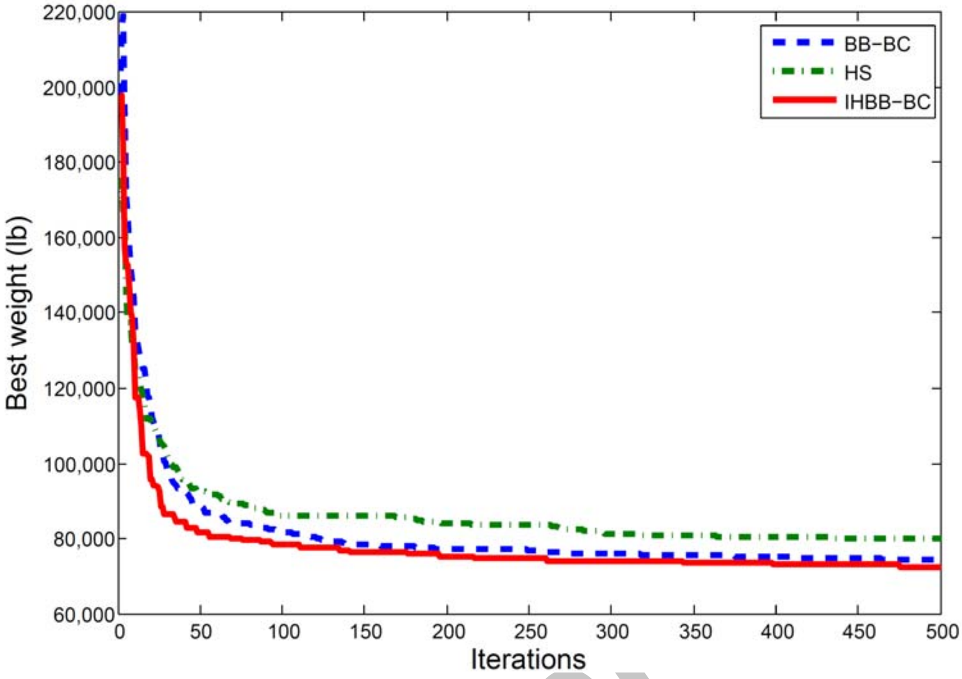

Figure 11. Comparison of the convergence history of the double layer braced barrel vault for three algorithms under symmetric loading

Considering Figure 12 it can be observed that some member groups of this structure have small stress ratios, and bigger cross sections are selected. The reason corresponds to the special configuration and loading type of this structure. Large vertical joint displacements at the apex of this structure results in a design which is governed by displacements. The stiffness of some members on the overall stiffness of the structure is another problem. In the process of optimization the stiffness of some members and hence the stiffness of the entire structure increases and such members are designed with high cross section areas. Selecting small sections for such groups of members will result in smaller weight, while selecting small cross sections for the members with axial compressive stresses (corresponding to buckling) and small displacements, the selection of high cross sections increases the weight of the structure. This problem is dealt with by selecting a generation with different solution candidates in the process of optimization. Also the problem of the flexibility of the large-scale structure is an important issue which is dealt with. In relation with this problem, parts of the structure which have low axial forces and displacements are designed by small cross sections and the other part because of having constraints are designed by large cross sections. Thus part of the structure will be stiff and other part will be flexible.

The uniformity of the distribution of stiffness in the vicinity of the structure is an important issue for large-scale structures. If part of the structure has elements of low axial forces and small displacements, and another part contains elements of high cross sections, then the uniformity of the distribution of the stiffness will not be achieved.

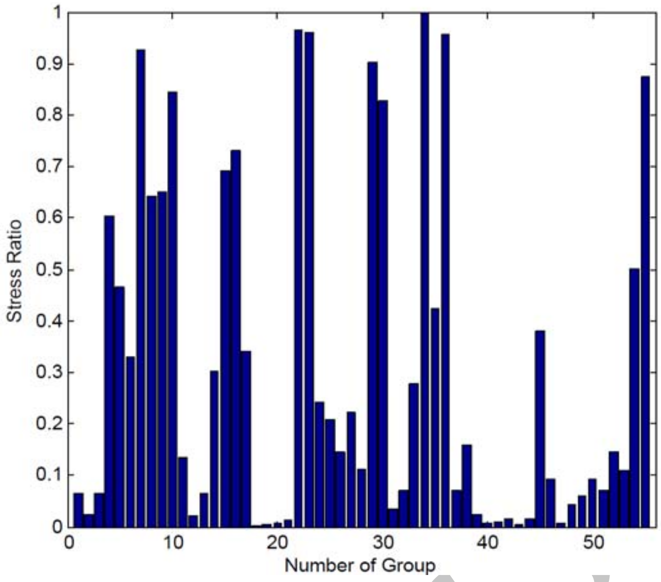

Figure 12. The ratio of the max stress to the allowable stress for different element groups for the double layer barrel vault for the IHBB-BC algorithm under symmetric loading

5. CONCLUDING REMARKS

Due to the special form of the barrel vaults, and because of its usage in practice, optimal design of this type of structures is very important. Here, the Big Bang-Big Crunch is applied for optimal design of truss type of barrel vaults.

The HBB-BC is an improved version of BB-BC developed by Kaveh and Talatahari [2], which uses the best global and best local candidate in the formation of generation. In this algorithm the Particle Swarm Optimization characteristics are added to improve the exploration ability of the BB-BC.

The IHBB-BC is a hybrid version of two improved optimization methods HBB-BC and HS. In this method increasing the number of iterations has additional effect on those element groups for which the cross sectional areas in the best global solution is near the boundary of the search space. This effect is due to the fact that if HBB-BC method creates solution out of search space for these element groups, then the solution will not lead to values near the boundary, while the use of HS method makes the algorithm to converge to the best global solution.

In the first example the structure has been under both symmetric and non-symmetric loading. For each group of elements the ratio of the maximum stress to its maximum allowable stress is calculated and shown in Figures 10 & 11. In some of these groups the zero force can be observed, and no stress is developed, and the ratio has its smallest value and the optimization process has led to the smallest possible cross sections. This can be observed from the tables of the cross sections.

For the second example, it can be seen from the diagram of the stress ratios and the corresponding tables for some group of elements though the stress ratio is small, however still some big cross sections are selected. In the first glance one may think that perhaps the optimization is not performed properly. However, investigating the entire process reveals that the penalty for the violations has become zero and the weight has become smallest possible, though some groups have selected heavy sections. Of course some other limitations have also role and for the element at the top of the structure the displacement are rather big and the displacements govern the design.

For evaluating the efficiency of the presented method, the results are compared to those of HS and BB-BC. One can easily see the improvement due to the use of IHBB-BC for truss type barrel vaults.

REFERENCES

1. Makowski ZS. *Analysis, Design and Construction of Braced Barrel Vaults*, Taylor & Francis e-Library, 2006, pp. 1-144.
2. Kaveh A, Talatahari S. Size optimization of space trusses using Big Bang-Big Crunch algorithm, *Computers and Structures,* **87**(2009) 1129-40.
3. Erol OK, Eksin I. New optimization method: Big Bang-Big Crunch, *Advances in Engineering Software,* **37**(2008)106-11.
4. Camp CV. Design of space trusses using Big Bang-Big Crunch optimization. *Journal of Structures Engineering ASCE,* **133**(2007) 999-1008.
5. Li LJ, Huang ZB, Liu F, Wu QH. A heuristic particle swarm optimizer for optimization of pin connected structures, *Computers and Structures,* **85**(2007) 340–9.
6. Kaveh A, Farahmand Azar B, Talatahari S. Ant colony optimization for design of space trusses, *International Journal of Space Structures,* No. 3, **23**(2008) 167–81.
7. American Institute of Steel Construction (AISC). Manual of Steel Construction Allowable Stress Design. 9th ed. Chicago, IL, 1989.
8. Saka MP. Optimum design of pin-jointed steel structures with practical applications, *Journal of Structural Engineering ASCE,* No. 10, **116**(1990)2599-620.
9. Lee KS, Geem ZW, A new structural optimization method based on the harmony search algorithm, *Computers and Structures,* **82**(2004) 781–98.
10. Mahdavi M, Fesanghary M, Damangir E. An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation,* **188**(2007) 1567–79.
11. Kaveh A, Talatahari S. Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. *Computers and Structures,* **87**(2009) 267–83.
12. Schutte JJ, Groenwold AA. Sizing design of truss structures using particle swarms. *Structural and Multidisciplinary Optimization,* **25**(2003) 261–9.
13. Kaveh A, Talatahari S. Optimal design of skeletal structures via the charged system search algorithm, *Structural and Multidisciplinary Optimization*, No. 6, **41**(2010) 893-911.